Aula 15

Roteiro

- Otimização
- Caixeiro viajante
- *Hill Climbing*
- Distribuição de Boltzman
- Simulated Annealing
- De volta ao caixeiro

Otimização

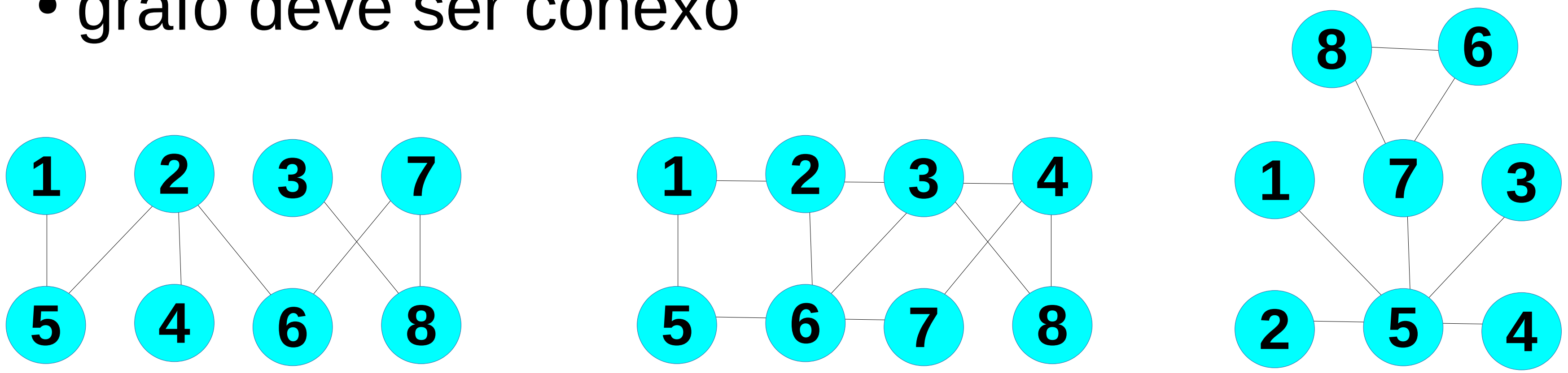
- Considere espaço discreto *S* e função *f* que avalia cada elemento, $f : S \rightarrow R$
- **Problema:** encontrar um elemento de *S* que minimiza (ou maximiza) *f*

$$S^o = \{ e \mid e = arg\,min_{s \in S} f(s) \}$$

- **Ideia 0:** enumerar elementos de *S*, encontrar valores mínimos
  - funciona apenas se |*S*| é pequeno (ex. 10^9)
- **Ideia 1:** criar uma *estrutura* sobre os elementos de *S*, procurar utilizando a estrutura
  - estrutura é um grafo, vértices são elementos de *S*

Exemplo

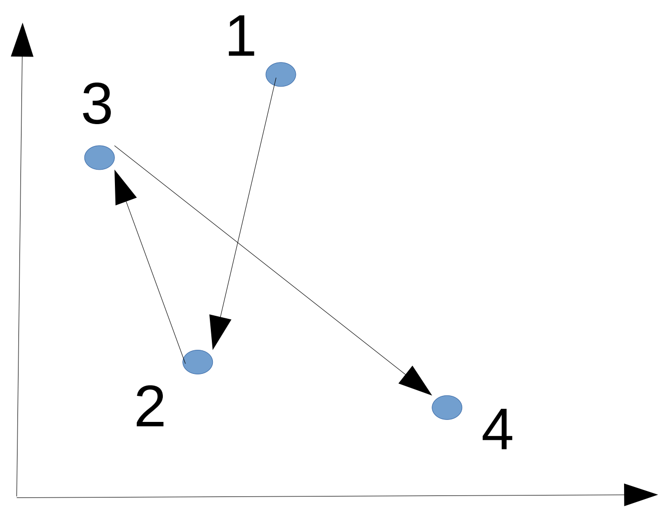
- $S = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $f(s) = 1/s$
- Criar um grafo com elementos de S
  - grafo deve ser conexo



- Usar estrutura do grafo para buscar por elemento que minimiza f
  - percorrer o grafo pelas arestas (busca em grafos)
  - usar valor de f nos vizinhos para guiar a busca

Caixeiro Viajante

- Considere *n* pontos no plano
  - $v_i = (x_i, y_i) \rightarrow$ coordenadas de cada ponto
- Considere a distância euclideana entre pares de pontos
- **Problema:** encontrar percurso com menor comprimento
  - percurso é permutação dos pontos
- *S* = todas as permutações; *f* = soma das distância entre pontos consecutivos na permutação
- Exemplo: $n = 4$

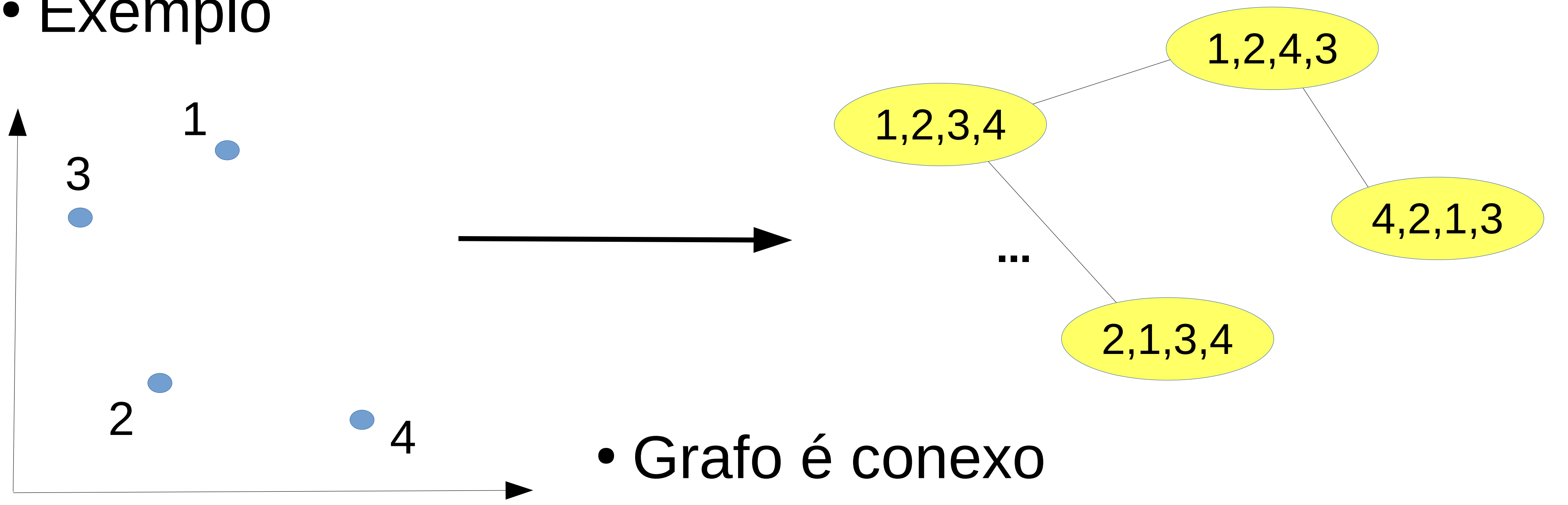


- $P_1 = 1,2,3,4$ $\quad f(P_1) = d(1,2)+d(2,3)+d(3,4)$
- $P_2 = 4,2,1,3$ $\quad f(P_2) = d(4,2)+d(2,1)+d(1,3)$
- $P_3 = 1,4,3,2$ $\quad f(P_3) = d(1,4)+d(4,3)+d(3,2)$
- Qual é o menor percurso?
  - número de percursos = $|S| = n!$

Figueiredo 2021

Grafo do Caixeiro Viajante

- Cada permutação é um vértice do grafo
- Aresta entre P_i e P_j sse P_i e P_j diferem em apenas um par de elementos
  - uma única troca entre as duas permutações
- Exemplo



- Grafo é conexo
- Cada vértice do grafo (permutação) define um custo: $f(s)$

Explorando o Grafo

- Como explorar o grafo em busca do ótimo?

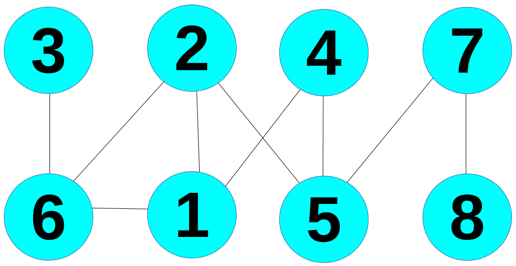
Algoritmo guloso

1) Começar em um vértice qualquer
2) Avaliar qualidade de todos os vizinhos
3) Transicionar para vizinho de menor/maior valor
4) Repetir enquanto puder

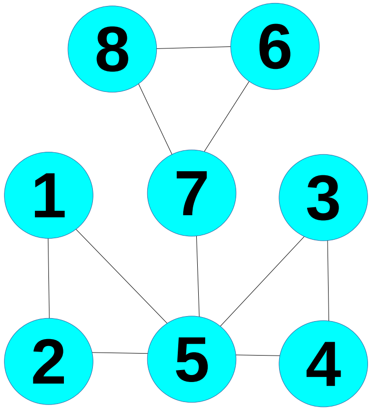
- Algoritmo conhecido como *Hill Climbing*
  - Desce/sobe gulosamente pela montanha
- **Problema:** mínimos/máximos locais
  - vértice cujos vizinhos são todos superiores/inferiores

Exemplo

- $S = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $f(s) = 1/s$
- Encontrar o mínimo
  - *Hill Climbing* → transicionar para vizinho de menor valor



- Iniciar em 4: 4 → 5 → 7 → 8
- Iniciar em 1: 1 → 6 (mínimo local)
- Iniciar em 2: 2 → 6 (mínimo local)



- Iniciar em 1: 1 → 5 → 7 → 8
- Iniciar em qualquer vértice leva ao mínimo global

Estrutura do grafo possui papel central!

Figueiredo 2021

MCMC to the Rescue

- Como lidar com mínimos locais?
- **Ideia:** Não ser tão guloso; usar aleatoriedade para controlar a gula!
  - permite transicionar para vizinho que seja pior, incluindo pior que vértice atual
- Transformar o grafo em uma CM irredutível e aperiódica
  - garante que todos os estados serão visitados
- Mas com qual distribuição estacionária?

π_s inversamente proporcional a $f(s)$

  - vértices com valores menores tem mais chance de serem visitados
- Usar *Metropolis-Hastings* para construir CM com π_s

Distribuição de Boltzman

- Espaço *S* e função *f* que avalia cada elemento
- Considere parâmetro $T > 0$, chamado de temperatura
- Probabilidade associado ao elemento *s* dada por

$$\pi_s = \frac{e^{-\frac{f(s)}{T}}}{Z}$$

Constante de normalização

$$Z = \sum_{s \in S} e^{-\frac{f(s)}{T}}$$

- Para um *T* fixo, menor *f(s)* maior π_s
  - valores menores de *f*(*s*) tem probabilidade exponencialmente maiores
- Para problema de maximização, trocar sinal da exponenciação

Distribuição de Boltzman

- O que acontece quando T é muito pequeno?

$$\pi_s = \frac{e^{-\frac{f(s)}{T}}}{Z}$$

Aumentantam as diferenças entre π_s

- valores menores se destacam mais

- Seja $\alpha(T)$ a probabilidade de um elemento de S escolhido aleatoriamente com probabilidade π_s ser mínimo em f
- Teorema: $\lim\limits_{T \to 0} \alpha(T) = 1$
- Boa notícia: probabilidade de escolher um elemento mínimo vai a um quando T vai a zero
  - amostrar CM usando π_s como distribuição estacionária
  - diminuir T para aumentar chances de escolher o mínimo

Simulated Annealing

- Técnica de otimização para resolver problemas de máximo/mínimo global em espaços discretos
- Construir uma sequência de CM usando *Metropolis-Hastings* e distribuição de *Boltzman* com diferentes valores para T
  - cada valor de T induz uma CM e uma distribuição π_s
- Escolher $T_1 > T_2 > T_3$ … com $T_i \rightarrow 0$ quando $i \rightarrow$ infinito
- Escolher N_1, N_2, N_3, …. e estado inicial qualquer X_0
  - N_i : número de passos na CM com temperatura T_i
- (T_1, N_1), (T_2, N_2), ... é chamado de *annealing* (*agenda de resfriamento*)

Algoritmo

- Simular CM com T_1 por N_1 passos a partir de X_0
  - seja X_{N1} o estado final
- Trocar para T_2 e simular CM por N_2 passos a partir de X_{N1}
  - seja X_{N2} o estado final
- Trocar para T_3 e simular CM por N_3 passos a partir de X_{N2}
  - e assim por diante
- Troca de temperatura muda apenas as probabilidades de transição não nulas (estrutura se mantém)
- **Ideia:** "resfriar" simultaneamente com a geração de um caminho amostral
  - torna-se mais difícil transicionar para valores piores
- Guardar estado de menor valor ao longo de toda simulação

Simulated Annealing

- $(T_1, N_1), (T_2, N_2), \ldots$ chamado de *annealing*
- Escolha do *annealing* é fundamental para garantir convergência para ótimo global
  - se resfriar muito rápido, CM pode "ficar presa" em mínimo local
- **Teorema:** se T_i decresce devagar o suficiente, então P[X_t ser ótimo] $\rightarrow$ 1 quando $t \rightarrow$ infinito
- **Problemas**

  (1) devagar o suficiente depende do problema

  (2) devagar o suficiente pode ser muito lento para ser usado na prática

Estratégias de Resfriamento

- Como T deve decrescer com o número de passos na CM?
- **Ideia:** definir temperatura para cada passo
  - $N_i = 1$ para todo i
- $T(t)$: temperatura a ser usada no passo t
- Funções geralmente usadas, para um $T_0 > 0$ e $0 < \beta < 1$

$$T(t) = T_0 \beta^t$$

Exponencial

$$T(t) = T_0 - \beta t$$

Linear

$$T(t) = \frac{a}{\log(t+b)}$$

Logarítmico

- Prova de convergência global se a for grande o suficiente e b constante
- Mas caminho amostral pode ser muito longo (t tem que crescer muito para temperatura ser baixa)

Estratégias de Resfriamento

- Estratégia de resfriamento anteriores era fixas
  - não dependem das amostras geradas
- **Ideia:** Estratégia de resfriamento adaptativa (dinâmica)
  - usa valores das amostras para resfriar
  - usar diferença $f(X_t) - f(X_{t-1})$ para definir redução em T (ex. proporcional a diferença)
- Adiciona camada de complexidade, na teoria
  - sequência de CM depende do caminho amostral
- Na prática, pode funcionar bem
  - sem muitas garantias teóricas

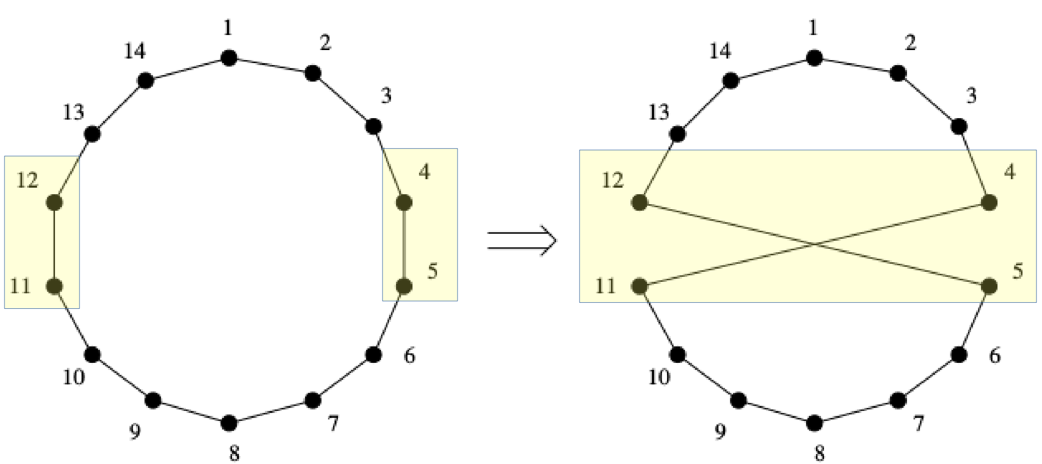
Voltando ao Caixeiro Viajante

- Permutações das cidades como estados da CM
- Transição entre permutações: inverter parte da permutação
  - escolher índices *i* e *j* em [1, *n*], com *i* < *j*, inverter permutação atual entre os índices *i* e *j*
- Exemplo

1 2 3 4 5 6 7 8 9 $\xrightarrow{i\,=\,3,\ j\,=\,6}$ 1 2 6 5 4 3 7 8 9

8 1 7 2 5 9 7 3 4 $\xrightarrow{i\,=\,2,\ j\,=\,4}$ 8 2 7 1 5 9 7 3 4

- Objetivo é redefinir (inverter) pares de arestas



- Exemplo com n=14
  i=5, j=11

Figueiredo 2021

Voltando ao Caixeiro Viajante

- Número de transições de saída = número de escolhas para *i*, *j*, com i < j
  - escolha sem repetição em [1, n] → n(n-1)/2 possibilidades
  - não depende da permutação
- Logo, P[(i, j)] = 1/(n(n-1)/2) para qualquer i, j
- CM base é simétrica
  - todo estado tem o mesmo grau de saída e mesmas probabilidades de transição
- Definir π_s de acordo com distribuição de Boltzman
  - *s* é uma permutação, com comprimento de caminho dado por *f*(*s*)
  - *f*(*s*) não depende de *T*, mas π_s depende

Voltando ao Caixeiro Viajante

- Definir CM com π_s via Metropolis-Hastings, com $T > 0$
- Probabilidade de transição de *s* para *s'*
  - CM base é simétrica
  - escolha de (i, j) define a transição (define quem é *s'*)

$$P_{s,s'} = \frac{2}{n(n-1)} \; min\left\{ e^{\frac{f(s)-f(s')}{T}}, 1 \right\} \quad , \quad \text{se s != s'}$$

- Se *f*(*s'*) for menor, então aceita com probabilidade 1
  - escolha uniforme entre vizinhos *s'* melhores que *s*
- Se *f*(*s'*) for maior, aceita com probabilidade que é inversamente proporcional a diferença
  - menos chances de aceitar permutação que aumenta muito o comprimento do caminho

Voltando ao Caixeiro Viajante

- Como definir agenda de resfriamento?
  - $(T_1, N_1), (T_2, N_2), (T_3, N_3), \ldots$
- Não temos muita teoria para isto (infelizmente)
  - usar estratégias anteriores
- Na prática, tentativa e erro usando experiência adquirida
  - começar com instâncias pequenas
- Exemplo interativo na web: http://toddwschneider.com/posts/traveling-salesman-with-simulated-annealing-r-and-shiny/

Ainda é tema de pequisa em diferentes áreas (física, computação, matemática, engenharia, etc)